the
AARDVARK JOURNAL
FEBRUARY 1982 VOL 2 NO. 6

*** INDEX ***

ARTICLE PAGE #

CHEAP DISK FOR C1P (SA400)........ 2-5

REVERSE VIDEO FOR C1P............. 6

RS232 FOR THE C4P................. 6-8

CHEAP & SIMPLE SERIAL INTERFACE ON MX-80....................... 9

TRANSFERRING DATA BAUD ON PRINTER by Billy D. Smith - letter..... 10

ABOUT THE OLIVETTI PRINTER by Phillip Brown............... 10

ENABLING ROM TO DISK by Don Chochola................ 10,11

PROGRAM, SAVING MACHING LANGUAGE L.A. Beer...................... 11

AARDVARK CLASSIFIED............... 11

ANOTHER LOOK AT OS65D V3.3 by Richard Tretheway........... 12

LINKING MACHINE CODE SOURCE FILES by Dave Edson.................. 13

PASSWORD BUSTER PROGRAM........... 13,14

DUMB TERMINAL PROGRAM............. 14

REPLACING NORMAL OUTPUT FOR A BLIND PERSON, by P. McCan...... 15

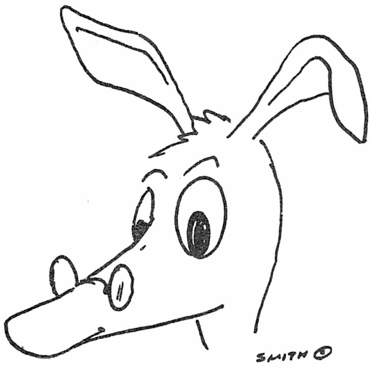
CORRECTION: C. SCHALL ON EXTENDED MONITOR PROGRAM.... 15

MEMORY MAP, by Bobby Akins........ 16

BAUD RATE SELECTOR, by D. Broyhill 16,17

YACHT RACE GAME LISTING........... 18,19

JOURNAL SUBSCRIPTION RENEWAL FORM. 20



** LOOKING AT NEXT YEAR **

We have decided to go on for another year of the journal - with a few changes.

The journal has been a money losing proposition that we do mainly for the fun of it - (and for the advertising value). As a result, it has a low priority on our time here. I like it, but we have to keep the doors open in order to be able to do a Journal. As a result, it is usually late - up to six weeks late. The bad news is, therefore, that we are going to raise the price to the point where the blasted thing at least breaks even. Volume #3 will cost $14.00 in the States and Canada, and $19.00 overseas. That should allow us to hire a little extra help to get the Journal out on time.

We are going to continue the technical articles as much as possible next year, but we are also going to take a step backwards. It occured to me as I looked at the last couple of issues that we are neglecting the beginner and novice. we are therefore going to re-establish the Beginners Corner - starting next month with an article on what Peeks and Pokes are good for. We have also contracted with the British Users Group to publish a couple of articles from them on machine code for beginners.

The format for the Journal will otherwise stay about the same. It doesn't look very impressive, but it contains more information on OSI than any other publication on the market and will continue to do so.

OSI AND THE SHUGART SA400

BY STANLEY WINDES

In recent months ads have appeared in leading computer magazines offering the Shugart SA400 5-1/4 inch minifloppy disk drive at a reasonable price of approximately $150 for a used unit. This started me thinking about a mini floppy for my Superboard-C1P. After some preliminary checking, I decided it could be done, and that it wouldn't be hard. I therefore sent my order on its way, and meanwhile I started digging in earnest for information. My first step was the Sams Photofact for the C1P. My ground rule was to use standard OSI software, meaning that my hardware was dictated by the OSI 65D operating system and must be functionally equivallent to that provided by OSI.

The following paragraphs describe the way I made the Shugart SA400 work with my Superboard. The following topics are covered in the order given:

A. Power Supply
B. Data Separator
C. Interface Board
D. controller to Disk connections
E. HEXDOS
F. OSI 65D

A. POWER SUPPLY

The power requirements of the SA400 disk drive alone are:

+12volts at 1.8amps. max. 0.9amps. typical
+5volts at 0.7amps. max. 0.5amps. typical

The power supply I built to supply these voltages is shown in Figure #1. The power Transformer is a Radio Shack 12volt at 3amp unit with approximately 20 turns of #18 enameled wire added to increase the voltage by 5volts. The rest of the parts are standard. The mechanical layout and mounting in relation to the disk is shown in figure #2. The electrical connections to the disk circuit board are shown in figure #7.

B. DATA SEPARATOR

The first thing that became apparent upon examining the OSI hardware was that the Data Separator was not on the 610 Board and it was not part of the SA400 electronics either. After much searching and head scratching I concluded that since the OSI system never misses inserting the clock pulse before each data pulse that is written on the disk, the Data Separator would be easy and simple to design. Some of the disk writing systems withhold some of the clock pulses written on the disk. This makes the data separator much more difficult. The data separator I came up with, and found to work very well, is shown in figure #3. The RC time constant is adjusted to give an output pulse width on pin#1 of the 74LS121 of 5.5 ls. The adjustment proceedure is: with disk disconnected and the controller connected to computer which is turned on and operating in the non disk mode, place a jumper wire from "write data" output line (8) to "receive data input" (12). This series of clock pulses will trigger the 74LS121 so that the RC time constant can be adjusted for a negative pulse width of 5.5 ls. For some one who has a 610 board and wants to add a SA400 drive this data separator and a power supply would be all they need.

C. INTERFACE BOARD (CONTROLLER)

Refer to Figure #4 during the following discussion.

The circuit I am using is basically the OSI circuit with modifications. First of all I already have 40K of memory up and running, therefore the buffered address lines were already available to me. For those of you who are willing to use HEXDOS (see later) additional (greater than 8K) memory is not required, but buffered address lines (use 75LS241) are almost a must.

I did not use the high current drivers to the disk since my ribbon cable is less than 18 inches long. Also since I had only one disk I decided to start simple and only provide for 1 drive. Once this is working a second can be added as required. After looking into the use OSI makes of the real time hardware interface to the 6821 PIA I decided not to implement this at the start. It can be added when needed. The address decoding I used (74LS30) is easier to implement and to understand. These points are what brought about the interface (controller) design that I show in figure #4.

The two sets of pin numbers on the right hand side of figure #4 are: The inner ones (board IC) are the pins of a 16 pin dip socket I used as a plug and socket for the cable to the disk. The outer ones labeled DISK J1 are the pins of the disk circuit board connector J1.

I would strongly recommend that the motor control circuit in the August AARDVARK Journal by Dave Pompea be incorporated. See figure #5 on how I added it to my decision.

The data direction control circuit which controls the Data buffers on the 610 board is shown in figure #6. The number of inputs you need to provide will naturally depend on your hardware configuration.

The adjustment of the 74LS123 one shot timers in the transmit data and receive data circuit is well covered in the Sams Photofact for the 600 and 610 board.

D. CONTROLLER TO DISK CONNECTIONS

The OSI 610 board provides several outputs to the disk drive which are not used by the 5-1/4 inch drive. Table #2 gives all the OSI output connector functions and the corresponding SA400 J1 pins. This information is provided for those who will be adding a SA400 to an existing OSI 610 board.

Also figure #7 is a sketch of the electronics board of the SA400 showing several areas of interest.

Before I get into the software aspects of this project there are several points that need be made. The SA400 has the ability to change the meaning of the write protect notch. That is, an open notch can be protected or not protected depending on the state of a application. The 5-1/4 inch disks I have received from AARDVARK have used the open notch to indicate a write protected disk. Therefore for these disks the "WP" trace located as shown in figure #7 must be cut or open. If you should desire a closed notch to indicate write protection then the "WP" trace must be closed. In addition I would recommend that the write enable line to the SA400 drive (J1-24) be disconnected from the controller and tied to +5V thru a 1K resistor until your system is debugged and you can successfully boot the system. This may save your disks from disaster. Also never turn the power off with a disk in the drive.

There are two things about the SA400 drive that makes it appear different to the OSI 65D operating system. The track to track stepping time for the OSI furnished MPI drived is 5ms. While the SA400 has a track to track stepping rate of 40 ms. As I show in the software discussion this is easily taken care of in the software. Second the number of tracks on a disk is also different. The OSI MPI drive has 40 tracks available while the SA400 has 35 tracks available. Again this is just a software correction. It is not required that this be changed (I haven't found any problems) but I will give the addresses to change later.

E. HEXDOS

The operating system I found best suited for making the hardware operational was HEXDOS by Steven P. Hendrix. This is an excellent operating system for the small memory user and also a very good choice for use when debugging hardware. HEXDOS is only one track long and uses the C1P monitor and basic ROMS. Since HEXDOS can complete its boot cycle by using only track zero we do not have to worry about disk stepping rates until after the boot is completed. Once the boot is successful, the software stepping rate can be changed to make HEXDOS work properly with the SA400. This is very easy to do. After booting up just hit break and then M, then go to address 04E1 for HEXDOS 2.3 or 04EB for HEXDOS 2.4 and write in a HEX 25. Then hit break and warm start and now HEXDOS is completely modified and all commands work. You can also use HEXDOS to initialize a new blank disk. This will write track zero on the disk with the new stepping rate. You now have a new bootable master with the correct stepping rate.

F. OSI 65D

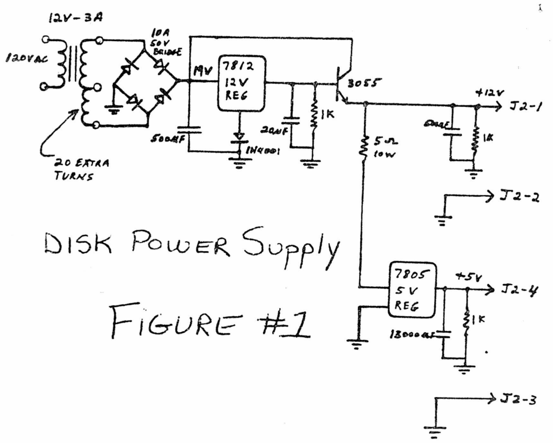
Now that my disk system was up and running on HEXDOS, I decided to go on to my final objective of running with the OSI 65D operating system. I obtained from AARDVARK their Fantastic Copy, SuperDisk II and Maxi-pros disks. Here again HEXDOS was invaluable. After booting up on HEXDOS read track zero of SuperDisk II into the memory, starting at address 8704($2200) using HEXDOS. Then going into the machine language monitor change location $26A3 to a $20. Then go back into HEXDOS and write memory starting at 8704 to track zero of a new disk. Then using Fantastic Copy, I copied the rest of SuperDisk II on to the new disk, thereby making a new master. The Fantastic Copy disk is really Fantastic and works great without modification. Should anyone not want to go the route of HEXDOS as I did, anyone with a C1 or C2 disk system can write a modified track zero on your SuperDisk II and then you will have a bootable master and you will be off and running. AARDVARK might be willing to supply a modified version of SuperDisk II.

I also wrote the modified track zero on the MAXI-PROS disk and the AARDVARK Galaxia game disk. Both work perfectly to date.

Also, as a further refinement, the locations of track zero and one shown in table #1 can also be changed to let the 65D operating system know the correct number of tracks for your disk.

One further note, the disks I obtained from AARDVARK are different in the subroutine called TENMS from the dissassembly obtained from SOFTWARE ASSOC. They say this software routine should delay for 10ms. Based on a value of $31 in location $267B and a 1mhz clock. The disks from AARDVARK have a $C7 in memory location #267B and therefore delay approximately 43ms. This may effect the disk drive compatability and any one planning to use a SA400 should check this value.

One last thought, this information is also useful to anyone who would like to add a second drive to their system and would like to do it cheap via a SA400. The only thing they would have to build is the data separator and power supply. The OSI provided drive will run at the slower stepping rate.



DISK Power Supply

FIGURE #1

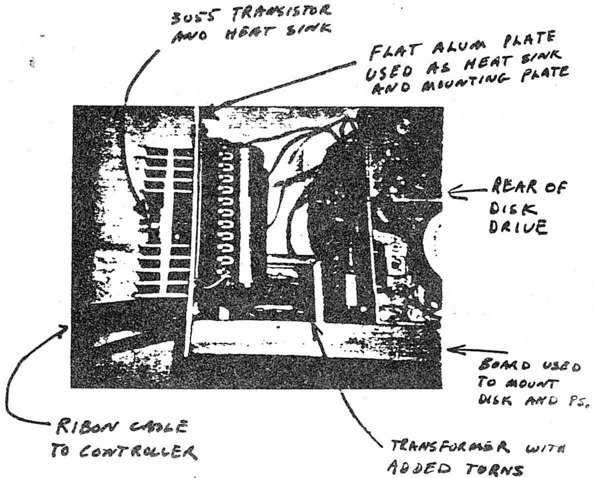
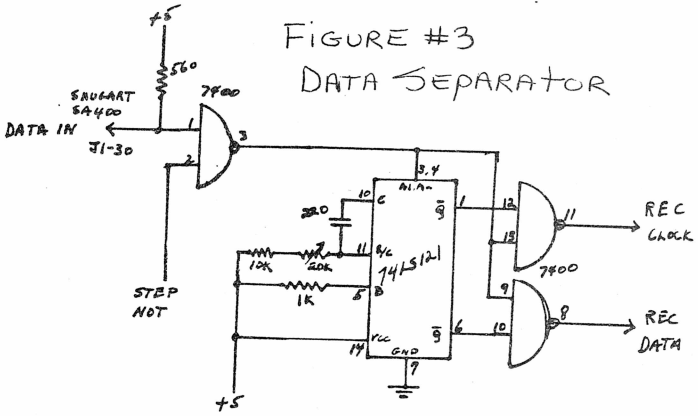
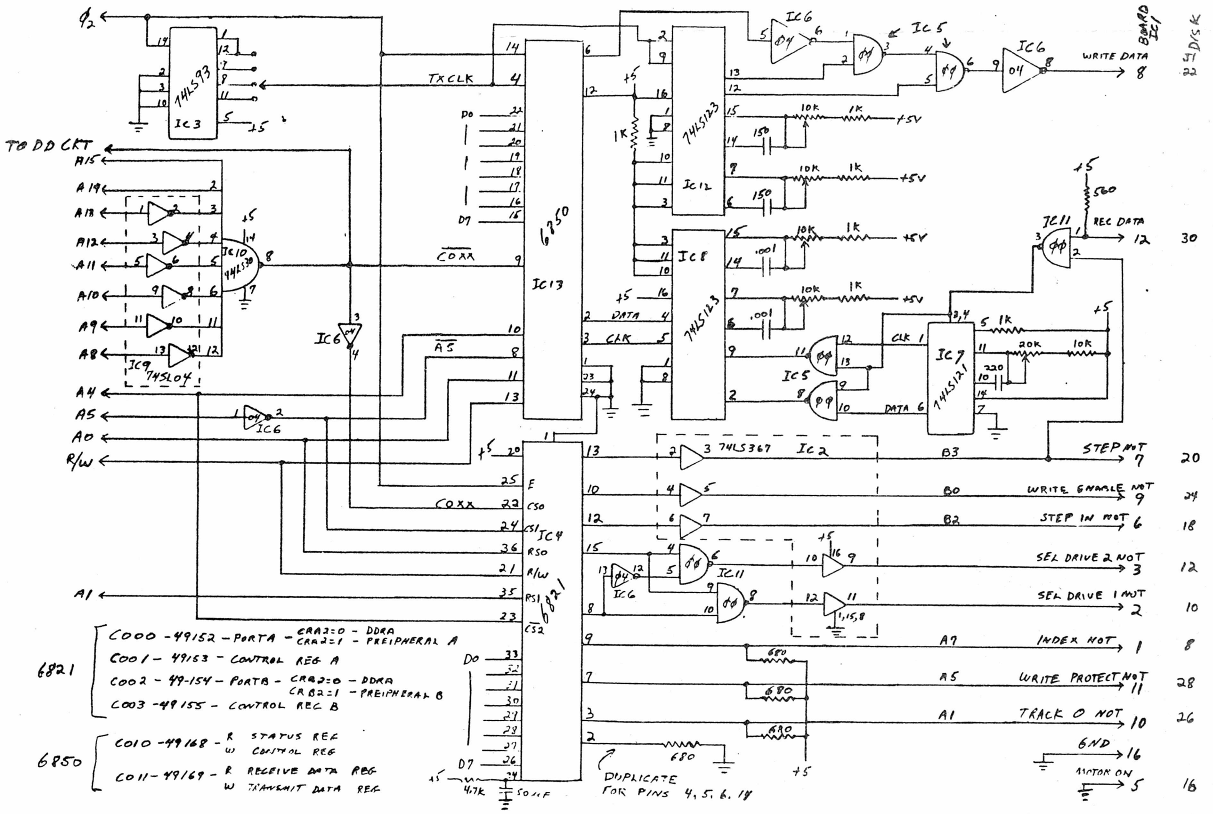

3055 TRANSISTOR AND HEAT SINK
FLAT ALUM PLATE USED AS HEAT SINK AND MOUNTING PLATE
REAR OF DISK DRIVE
BOARD USED TO MOUNT DISK AND PS.
RIBON CABLE TO CONTROLLER
TRANSFORMER WITH ADDED TURNS

FIGURE #2

FIGURE #3

DATA SEPARATOR

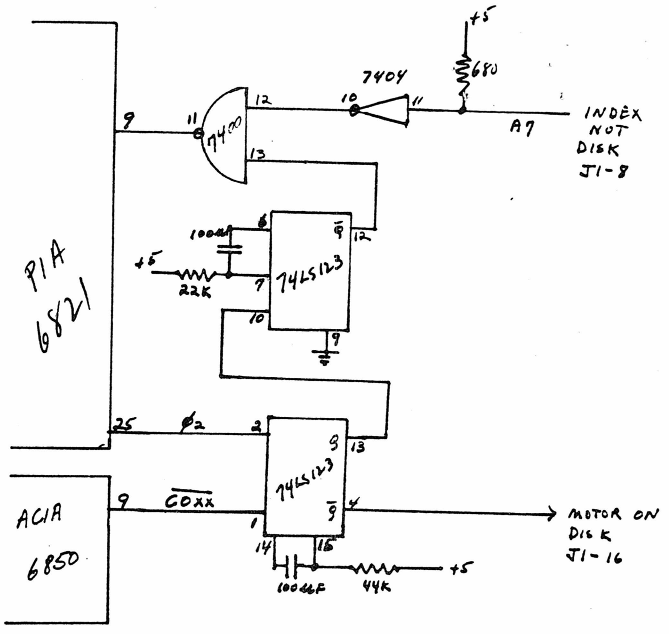




6821
C000 - 49152 - PORTA - CRA2=0 - DDRA
CRA2=1 - PREIPHERAL A
C001 - 49153 - CONTROL REG A
C002 - 49-154 - PORTB - CRB2=0 - DDRA
CRB2=1 - PREIPHERAL B
C003 - 49155 - CONTROL REG B

6850
C010 - 49168 - R STATUS REG
W CONTROL REG
C011 - 49169 - R RECEIVE DATA REG
W TRANSMIT DATA REG

FIGURE #4



MOTOR CONTROL
FIGURE #5

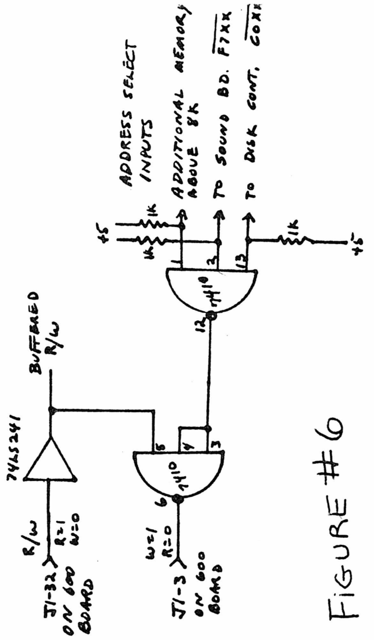
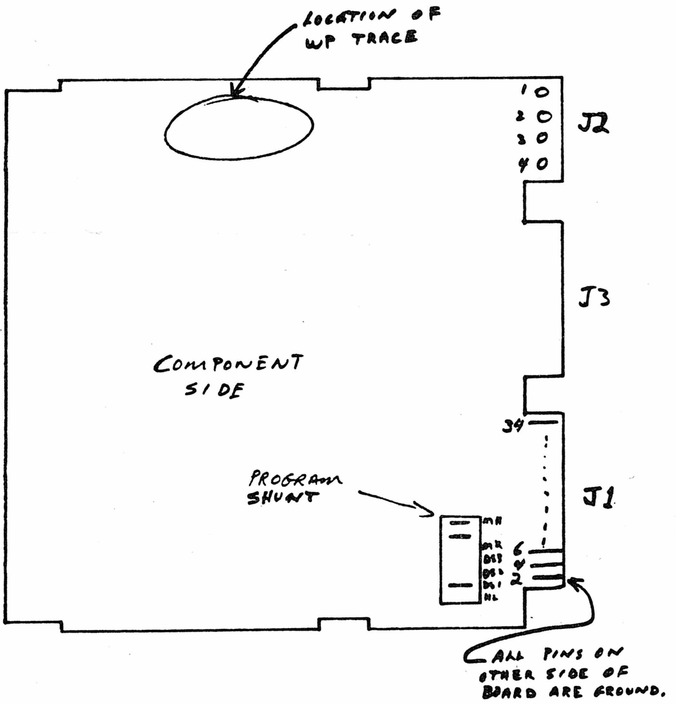


FIGURE #6

DATA DIRECTION CKT



DRIVE PRINTED CIRCUIT BOARD

FIGURE #7

DSI 610 BOARD CONNECTOR	FUNCTION	SHUGART SA400 CONNECTOR
J3-1	HEAD LOAD NOT	
2	LOW CURRENT NOT	
3	DRIVE SELECT #1 NOT	J1-10
4	FAULT RESET NOT (USED FOR MOTOR CONTROL BY DAVE POMPEA)	
5	STEP	J1-20
6	STEP DIRECTION	J1-18
7	ERASE ENABLE NOT	
8	WRITE ENABLE NOT	J1-24
9	WRITE DATA	J1-22
10	RECEIVE CLOCK	*
11	RECEIVE DATA	*
12	GROUND	ALL ODD PINS OF J1
13	GROUND	
14	NOT USED	
15	NOT USED	
16	NOT USED	
17	INDEX NOT	J1-8
18	DRIVE SELECT #2 NOT	J1-12
19	WRITE PROTECT NOT	J1-28
20	DRIVE #2 READY NOT	
21	SECTOR NOT	
22	FAULT NOT	
23	TRACK 00 NOT	J1-26
24	DRIVE READY NOT	
	MOTOR CONTROL	J1-16
	DRIVE SELECT #3 NOT	J1-14

TABLE #2

*SEE DATA SEPARATOR

HEX ADDRESS		CHANGE TO HEX VALUE
$26CA	TRACK 0	$35
$2669		$34
$2779		$34
$2DA7	TRACK 1	$34
$2DB7		$34

TABLE # 1

REVERSE VIDEO MODIFICATION

by David L. Kuhn

I know, I know...there are probably a thousand and one ways of doing reverse video. Here is one that I think is really neat, and is under software control. It only cost a couple of bucks to implement and will only work when turned on.

It uses the signal available on J75. J75 is a 16 pin dip socket, on the series II 600 board, that the 630 color board plugs into. Since I don't use color, I found that I could do a lot of neat things with this socket.

The signal lines that I use are the Color Enable (pin 13), Video (pin 10), and Video in (pin 11). The ckt is shown below. The exclusive OR gate is used as an inverter that is turned on and off with the Color Enable line (from here on out the C.E. line). If the C.E. is high the chip is an inverter. If it is low (as is when the computer first comes up), it acts simply as a buffer and you have normal video. By poking a 2 into location 55555 dec, you will bring up reverse video.

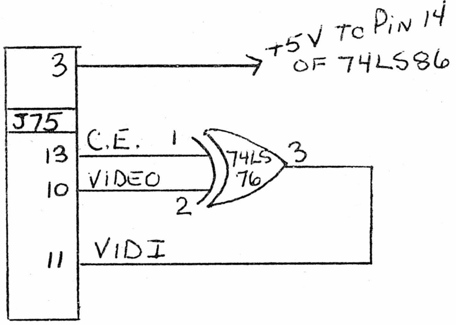
The modification requires that you cut the foil run between pin 9 of U42 and pin 2 of U70. Don't be afraid, according to my SB II series II schematic, it was meant to cut! This unshorts pins 10 & 11 on J75 and allows you to manipulate the video signal.

Now wire the circuit up as shown on the diagram. You could do it on a dip plug header from Radio Shack, and then have a small plug-in-modual. I did it on a breadboard set beside the computer, because I plan to experiment with all the signals that are there.

I hope that I have given some of you ideas. Besides, reverse video is great! It uses up no RAM and is great to play an AARDVARK adventure in the reverse video mode, and because it's so much nicer to read backwords!

TO USE:

```
POKE55555,2-
   REVERSE VIDEO (24X24)
POKE55555,0-
   NORMAL VIDEO (24X24)
POKE55555,3-
   REVERSE VIDEO (48X12)
POKE55555,1-
   NORMAL VIDEO (48X12)
```



DAVID RODENBERGER, COLUMBUS GROVE, OHIO

I am a systems and production Engineer for Triplett Meter Corp. in Blufton, Ohio. Recently I purchased a C4P computer, and after using it for a few weeks, I wanted to hook up my printer for hard copy. In trying to do so, I found that it wouldn't work. I called the store where I bought it and asked them about the problem. I was told that a printer couldn't be used on the cassette based system. This was very disturbing news. After studying the schematics I found that it would be very simple with a small amount of hardware to hook up the printer. All that is needed is a 14 pin IC socket, a 7404 TTLIC, a 2N5138 transistor, (2) 10,000 ohm resistors - 1/4 watt, (1) 470 ohm 1/4 watt resistor, and a negative 9 volt power supply.

The first thing is to remove the bottom cover plate by removing 6 screws. Also the ground wire will have to be removed. Place the computer upside down on a protective surface with the keyboard to your left. Remove the screw on the lower right side of PC board. Very carefully remove the board from the socket. It is a good idea to place your hand on the back PC board when you remove any of the boards so you won't crack it. Look at the IC layout diagram to locate IC U31. This is the TTl 7404 you have to install. I used a socket in case I ever have to trouble shoot the circuit. The manufacturer filled the holes in with solder, so you will have to remove it before installing the socket and associated points. When you have done this, install IC socket & solder into place (note: locator pin is facing up with the board positioned in such a way that the large row of connector sockets J1 are to your right). When you install the IC, pin (1) of IC will be on your left facing up.

Now locate the transistor Q2 in the pictorial. Note the location of collector Base & Emitter. After removing the solder from the holes install transistor and solder.

The three resistors are next. Again refer to the pictorial. The resistor mounting holes are just below Q2. The (2) 10K resistors are located on the outside with the 470 ohm in the middle. The solder will have to be removed from the holes before installation. Note take care not to use too much heat on the board and also observe the precautions for handling CMOS integrated circuits. Not all IC's on this board are TTL.

After you have completed the above reinstall the board and mounting screw. A short jumper will have to be installed from the PC board to the I/O board if you want to use the existing socket on the back of computer. If you don't have a socket like that, you can drill and mount a RCA style phono plug just above the audio jack on the back of the computer. The jumper will go from pin

7 of J3 on PC board to pin 3 of J8 on I/O board. The ground will be from pin 10 of J3 on PC board to pin 7 of J8 on I/O board. If a phono plug is used, the center conductor of the phono plug will go to pin 7 of J3 on PC board. Ground will be to pin 10 on J3 of PC board.

After all this is complete, you will need to install a negative 9volt supply. The negative output will go to pin 24 of the J1 on the 502 board. I have included a schematic of the power supply. The parts can be found in a junk box or found at any good Electronic Supply stor. You should have plenty of room to mount power supply in the large space by the computer supply.

To run the Printer, type SAVE. This will reflect or echo the keyboard. If you type LIST, your memory will be dumped on to hard copy. To turn off printer, type LOAD, space, RETURN.

If you have any questions feel free to call me. My phone number is (419) 659-5568 after 6:00.

I also have the information on how to transmit from printer to computer and on hooking up modems. If you need this information please contact me.

DAVID RODENBERGER, OHIO

If you want to send from the Printer to the Computer you will have to install another IC and Transistor. U20 and Q4.

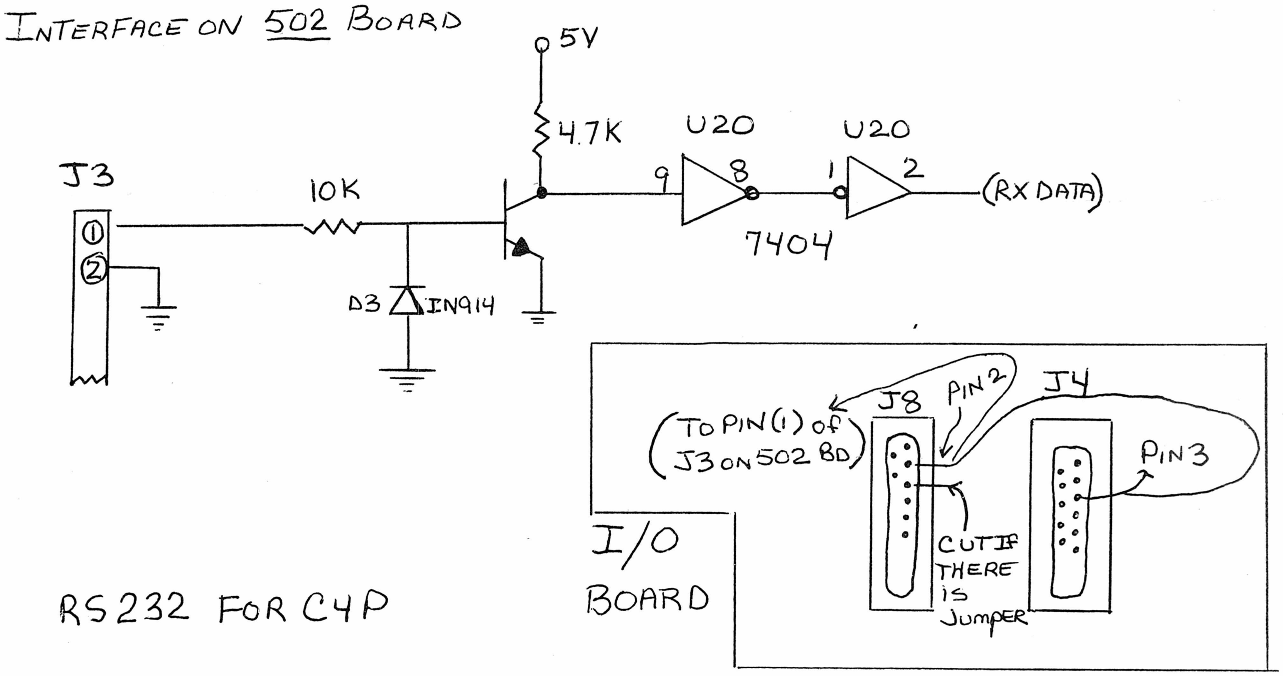
Locate U20 on the 502 board. Refer 70 IC layout diagram. Install and solder 14 pin socket in this location. You will have to clean out the holes before doing so. Now install IC U7404.

Locate Q4 on diagram. Clean out holes and solder into place. Watch Bosing. This is a 2N5138 transistor.

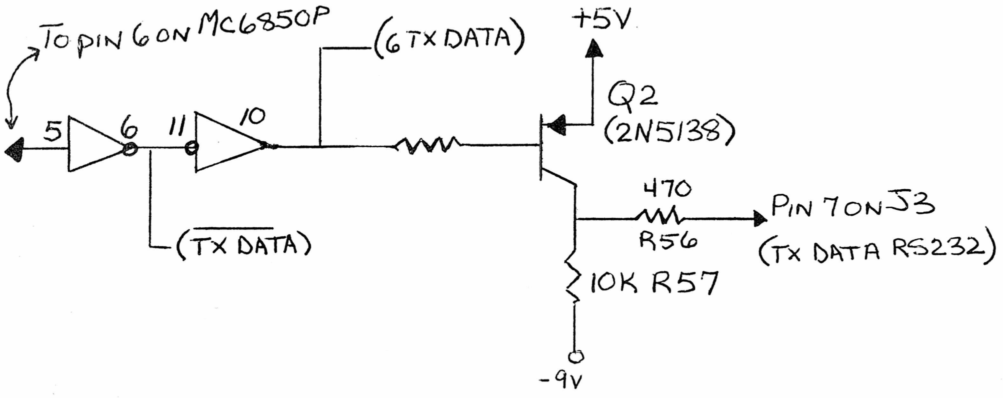
Refer to diagram and install resistors R61, R62. R61 is a 10K 1/4 watt and R62 is a 4,700 ohm 1/4 watt resistor.

Install D3 which is a 1N914 diode. All that is left is to run a wire from pin (1) J3 to pin 2 of J8 on the I/O board. Also run a jumper from pin 3 of J9 on I/O board to pin 2 of J8 on I/O board. Last of all check to see if there is a jumper between pin 3 and 2 of J8 on I/O board. If there is, cut it. You may now reinstall the 502 board and reassemble the computer.

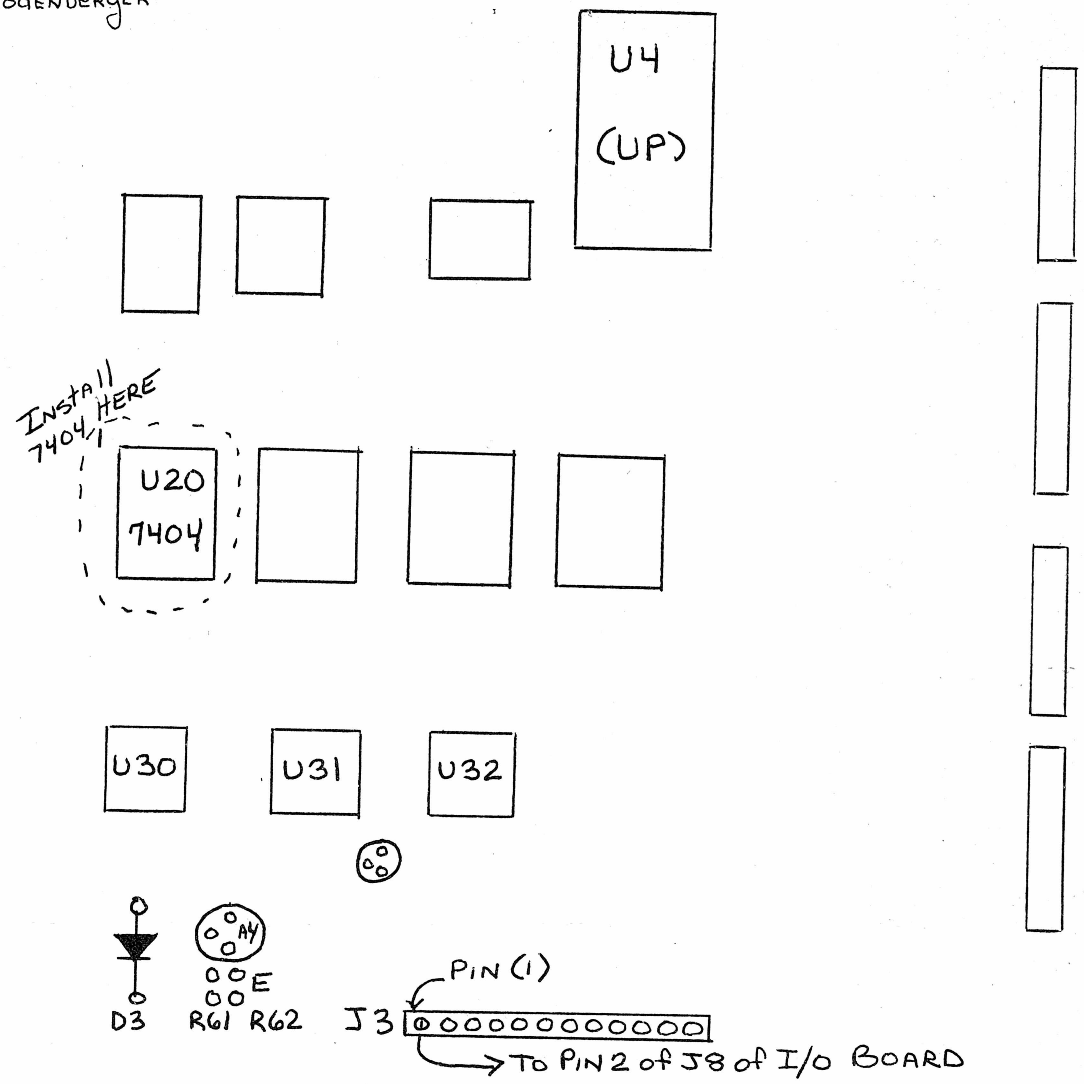
To use this function type Load, then return. To turn off type Space, return. Have fun.

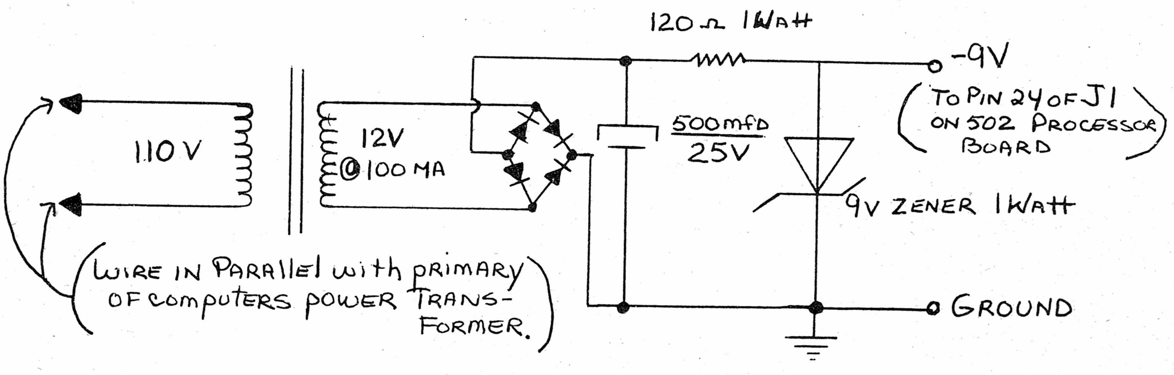


RS232 FOR C4P



D. Rodenberger

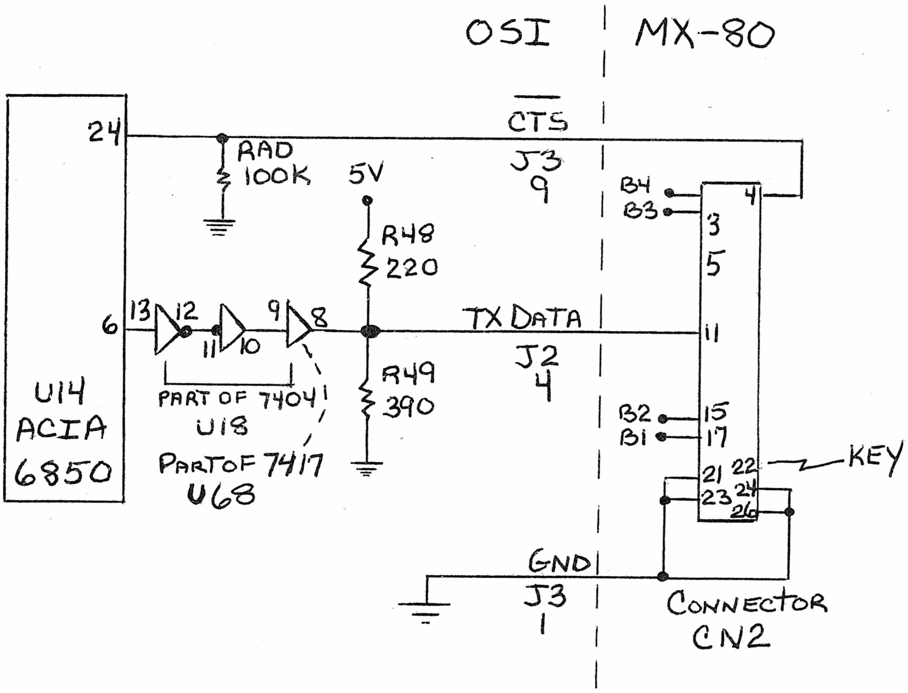



120Ω 1WATT
-9V
To PIN 24 OF J1 ON 502 PROCESSOR BOARD
110V
12V @100 MA
500mfD 25V
9V ZENER 1WATT
WIRE IN PARALLEL with primary OF computers power TRANS-FORMER.
GROUND

CHEAP & SIMPLE SERIAL INTERFACE FOR EPSON MX-80

ANONYMOUS, INDIANA

The Epson Serial/RS232 interface board is overkill in both capability and cost. A C1P/Superboard (and probably any other OSI) can be easily interfaced to the printer, since the needed capability is already built into the printer. The following is a schematic of the necessary interfacing for a C1P/Superboard:



MX-80 INTERFACE:

The interface is plugged into the CN2 connector located on the main PC board near the rear of the printer. The orientation of the connector can be determined by locating the key in pin 22. The connections can be made using wire wrap socket pins mounted through a perf bord with .100 spacing. The connections can be determined fromthe above schematic except for B1,B2,B3 & B4 which determine baud rate as below:

Baud	300	600	1200	2400	4800	9600
B1	G	G		G	G	
B2	G		G			
B3					G	
B4		G				

WHERE G = GROUND

If the cassette baud rate has not been modified, the printer should be operated at 300. It can be operated at 4800 if after any BREAK, use of the printer preceeded by setting up the ACIA with a POKE61440,3:POKE61440,16. If the cassette baud rate has been modified to 600 then the above techniques apply for 600 and 9600 baud printer use.

OSI INTERFACE:

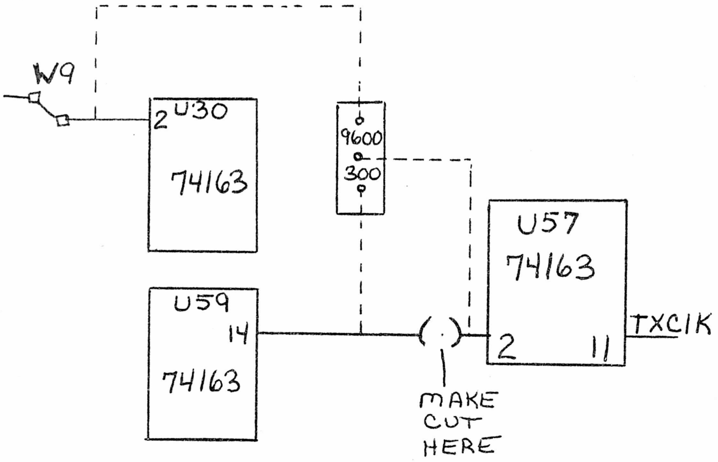
To enable the CTS pin on U14, the jumper to ground must be opened and a resistor RAD must be patched on to keep this pin low when the printer is not used. To enable the TXDATA output, the resistors R48 & R49 and IC U68 must be inserted into existing PC connections. IC U18 is already in place.

USE:

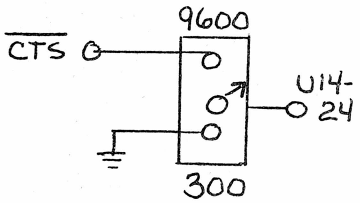
The software is already in place in the OSI for transmission of serial data to the printer and for interrupting this data stream anytime the CTS lead goes high. The printer is connected to the ACIA that normally drives the cassette, so it is turned on & off like the cassette, with SAVE & LOAD or POKE517,255 & POKE517,0. Once the printer is enabled, a listing may be obtained by a simple LIST, and a item printed by a simple PRINT.

BILLY D. SMITH, RICHMOND, KENTUCKY

After hooking up my MX-80 to my C1P disk, I realized that dumping listings and data at 300 baud was even slower than using a cassette based system due to the CTS line delaying the transfer of data to the 80's buffer. To overcome the lack of speed I bought the disk system to begin with, I made a simple mod to the C1P to transfer the data at 9600 baud, which is the maximum speed allowed with the MX-80 serial board. It only requires one trace cut and either a SPDT or DPDT switch and some wire to hook it up with.



A. Make cut at U57 pin 2. This isolates input to baud rate counter.
B. Connect center commonof switch to U57-2.
C. Connect one side of switch to U59-14 (other side of out). This is 300 baud select.
D. Connect other side of switch to U30-2. This is 9600 baud select.
E. Since the CTS line is only used in 9600 baud position, a DPDT switch can be used to control that line.



PHILLIP K. BROWN, SAN FRAN., CALIFORNIA

This may be an unfair and stupid question, but as I am brand new to the computing game I thought I would ask. Have you had any experience or have you heard or do you have any information about interfacing an Olivetti as a printer. In the May issue of Byte Mag. there is an add from a company called Vertical Data Systems Inc of Canada that says they have a simple plug-in module which converts Olivetti electronic typewriters into printers. Do you have any knowledge or information about this?

I have a Superboard II and if I were to do this I realize I would need more memory and a disk, etc. But it is an interesting idea, to be able to use a $700 electronic typewriter as a printer for letters, etc. and any help you could give would be appreciated.

I subscribe to your journal and I must say that I completely enjoy it and look forware to receiving it.

DEAR MR. BROWN,

Sorry, no one I know has succeeded in interfacing an Olivetti typewriter to an OSI computer. I never heard of Vertical Data systems, but that's probably because I never read Byte Mag. If you succeed in finding anything out about it, let us know. I think our readers would be interested.

A disk is not absolutely necessary for the use of a printer. The main problem is that it is impossible to store information on tape at 300 baud and with a decent word processor you can only get about 1 page at a time in memory when you have 8K. You can, however, do program listings and some limited text editing such as individual letters with the system that you have now. Your system does, of course, contain a 300 baud RS232 interface to run a printer.

DON CHOCHOLA, BOLING BROOK, ILLINOIS

I am the owner of a C2/4P. After about 3 years of fumbling with cassettes I decided to go to a disk based system. I just finished my disk interface board and tried to boot a disk using Cegmon boot program. I've found that the following changes are necessary to the ROM to enable the disk (Shugart 800). The interface board was supplied from D & N Micro in Ft. Wayne, IN.

cont'd

CEGMON	CHANGES NECESSARY	
F703 LDY#$00	F70C LDY #$80	Set PIAADD registers
F70E STY$C001	F70E STY $C001	
F711 STY$C000	LDY #$40 STY #$C000	Set for drive #2
F71C DEY	F71C LDY #$FF	Set PIABDD registers
F731 LDA#$FF	LDA #$2F	Select drive #1

As you can figure out this change takes more bytes than the ROM has at those locations. I have relocated the entire program with the changes and find that now I can access to track 0 on drive #1. I have no idea as yet concerning the program to load from track 0 so I can load and save to disks. I still have to expand the memory of my system (only 8K) to handle OS65D, but when I purchase 65D will the rest of the boot program in CEGMON be able to load into memory from the disk? YES

Also, when a cassette at 1200 baud the following trick works for BASIC. Clock at 2 meg hz.

```
POKE13,40:POKE518,20:REM EXTRA
NULLS FOR BASIC, SPACE BETWEEN
CHARACTERS USING VIDEO BAUD
SIMULATOR.
```

If the BASIC line is more than 64 characters long, then that line and the line after get garbled. CEGMON loads and saves machine code tapes at 1200 baud with no problem.

I have a 502 board in my C2/4P and noticed that it is identical to the 502 board used in the 4P. The video board is a 540; although not the same as the 540 used in the 4P, it has the hardware for a cassette interface. Using the ACIA on the 502 to drive the cassette via the 540 board has enabled a second possible output from the 502 board. The C4P service manual shows how the extra port can be used.

FOR A SUPERBOARD OR C1P

by L.A. Beer, California

Here is a BASIC program to SAVE machine language program code on tape, and a companion program to LOAD the tape anywhere in random memory. It is slow, but reliable. It will copy from anywhere in ROM or RAM and will load anywhere in RAM. The START and FINISH addresses asked for by the programs are in decimal. Hit RETURN immediately after entering 9 to signify "ready".

In the SAVE program, the twenty 99's used to be sure the tape is in sync are probably excessive. Five would be plenty. The 88 is to tell the program to start saving the machine code. In the OAD program, the 88 tells the program to start POKEing what it reads from tape.

In addition to saving your own hand-written machine code, you can "lift" routines from ROM to include in your programs.

```
LIST100-190

 100 FORN=1TO20:PRINT:NEXT
 105 PRINT"PROGRAM TO SAVE MACHI
NE LANGUAGE ON TAPE"
 110 PRINT"WRITTEN BY LOU BEER"
 115 FORN=1TO10:PRINT:NEXT
 120 INPUT"START ADDRESS";S
 125 INPUT"FINISH ADDRESS";F
 130 PRINT:PRINT:PRINT"START REC
ORDER-KIT 9 KEY WHEN READY"
 135 INPUTR:IFR=9THEN145
 140 GOTO135
 145 SAVE
 150 FORM=1TO20:PRINT"99"
 155 NEXTM
 160 PRINT"88"
 165 FORN=STOF
 170 PRINTPEEK(N)
 175 NEXTN
 180 POKE517,0
 185 PRINT"RECORDING FINISHED"
 190 END

OK
```

```
LIST300-395

 300 FORN=1TO20:PRINT:NEXT
 305 PRINT"PROGRAM TO LOAD MACHI
NE LANGUAGE FROM TAPE SAVED IN B
ASIC"
 310 PRINT"WRITTEN BY LOU BEER":
FORN=1TO10:PRINT:NEXT
 315 INPUT"START ADDRESS";S
 320 INPUT"FINISH ADDRESS";F
 325 PRINT:PRINT:PRINT"START CAS
SETTE WITH TAPE TO BE LOADED"
 330 PRINT:PRINT:PRINT"HIT 9 KEY
 WHEN READY"
 335 INPUTR:IFR=9THEN345
 340 GOTO335
 345 LOAD
 350 INPUTG:IFG=99THEN360
 355 GOTO350
 360 INPUTG:IFG=88THEN370
 365 GOTO360
 370 FORN=STOF
 375 INPUTD:POKEN,D
 380 NEXTN
 385 POKE515,0
 390 PRINT"MACHINE LANGUAGE LOAD
ING FINISHED"
 395 END
OK
```

** NOTICE **

AARDVARK SELLS EXCESS OFFICE EQUIPMENT.

THIS EQUIPMENT WAS ALL IN WORKING ORDER WHEN TAKEN OUT OF SERVICE. IT WAS ALL USED FOR SOFTWARE PRODUCTION AND DEVELOPMENT AT AARDVARK.

C4P-MF 24K. This is my personal computer and is in excellent condition. The brand new price on this unit is currently $1995.00. This was burned in properly and in excellent condition and we'll sell it for $1395.00.

GOLD STAR MONITORS. These B/W units also function as television sets in their spare time. They currently sell for $119.00 from OSI. We have three to sell and they will go for $59.00 each.

EPROM BURNER. An assembled board from OSI designed for the C4-C8 48 pin buss series. This burns 2 and 4K eproms. It currently sells for $197.00 from OSI and I'm going to sell mine for a measly $99.00.

IDS-440 PAPER TIGER PRINTER. This is a high quality Dot Matrix printer that doesn't get much use around here. We paid $1295.00 for it, used it a little, and are now going to sell it for $595.00. It does have a serial interface and is set up already to interface with an OHIO Scientific system.

CALL AARDVARK (313) 669-3110

RICHARD TRETHEWEY, MINN., MINNESOTA

I just received my December 1981 Aardvark Journal. I thought it was getting to be about time for another, but I know how it is. Don't feel bad about it, CompuServe publishes a monthly newsletter called "UPDATE" for it's subscribers and the information in it is always out of date and it is also always sent 6 weeks after the date of the issue.

What I am writing about is the article in this journal on OS65D V3.3. As a former OSI employee and since much of 3.3 was written by a very good friend of mine, I get defensive about people taking potshots at it without really checking it out. It seems that it has become standard practice to accept any claim that OSI software has a bug as gospel when such is usually not the case. Since 3.3 figures to be the very last product that will be produced by MAOSI for personal systems, I think it behooves us to make sure that it is given the praise due it andnot to let cheap shots at it go unchallenged.

I understand that the author of the article, Mr. Steven Gale, is a newcomer to disk systems and as such was totally unaware of the indirect file feature of 65D and most people will realize that he accidentally hit a <CTRL> 'X' and got garbage scrolling past him, so I'll passon that one. His contention that the new keyboard poll requires the comma and colon string input terminators to be reset and also the null input feature to open disk files is wrong. See the following example to prove this. I would have expected you to catch that one, Rodger, as soon as he said that the keyboard poll was responsible for the alleged foible. Mr. Gale also neglected to mention that the 3.3 manual is written as a tutorial guide that takes the user by the hand from boot-up through all the various disk functions and commands in BASIC. In addition, there is a new updated PEEK and POKE list, a memory map, and hard copies of all of the utilities. I realize that Mr. Gale obviously wrote this article very soon after he bought 3.3 and that is probably why he glossed over the two extra manuals included with 3.3, they are the BASIC REFERENCE MANUAL and THE ASSEMBLER/EDITOR and EXTENDED MONITOR REFERENCE MANUAL. The BASIC manual cross references all version of OSI BASIC and the ASM-EM manual gives more detail about these two utilities than was ever available before. Two of my favorite programs from the 3.3 package were also just mentioned in passing. First, the new modem program utilizes the Hazeltine emulator for output so that many bulletin boards and such can do screen clears. My second favorite is the resequencer program. This program not only renumbers lines, but GOTOs, GOSUBs, and IF...THENs. The command "RSEQ" also allows specifying line ranges and increments. Last, but certainly not least, no mention was made of the fact that OS65D V3.3 is fully upward and downward compatable with all previous versions.

```
5 POKE 2972,13:POKE2976,13
10 DISK OPEN,6,"DATA"
20 FORN=1TO10:PRINT#6,N:NEXT
30 DISK CLOSE, 6
35 DISK OPEN,6,"DATA"
40 FORN=1TO10:INPUT#6,K:PRINTK:NEXT
50 DISK CLOSE,6

READY WHEN YOU ARE, SWEETHEART
RUN
1
2
3
4
5
6
7
8
9
10

READY WHEN YOU ARE, SWEETHEART
6 POKE2888,0:POKE8722,0
DISK!"IO ,02

5 POKE2972,13:POKE2976,13
6 POKE2888,0:POKE8722,0
10 DISK OPEN,6,"DATA"
20 FORN=1TO10:INPUTA$:PRINT#6,A$:NEXT
30 DISK CLOSE,6
35 DISK OPEN,6,"DATA"
40 FORN=1TO10:INPUT#6,A$:PRINTA$:NEXT
50 DISK CLOSE,6

READY WHEN YOU ARE, SWEETHEART
RUN
? YOU CAN INPUT STRINGS, ANY
? WAY YOU LIKE.
? EVEN IF:THEY  CONTAIN
? COMMAS
? COLONS
?
? NULL INPUTS
? OR ANYTHING
? ELSE
? THE ARTICLE IS BALDERDASH!
YOU CAN INPUT STRINGS, ANY
WAY YOU LIKE.
EVEN IF: THEY CONTAIN
COMMAS
COLONS

NULL INPUTS
OR ANYTHING
ELSE
THE ARTICLE IS BALDERDASH!

READY WHEN YOU ARE, SWEETHEART
DISK!"IO ,02
```

LINKING MACHINE CODE SOURCE FILES

By Dave Edson
Meridian, Idaho
(208) 888-4707

The computer I use at home is an OSI 48k (I ripped out the basic) C1P-MF. Right now I am writing an all machine code word proccessor that will run on tape or disk and look like SCRIPSIT. I ran out of memory last night. So, the only choice I had other than giving up functions was to link my program.

You have the following source code:

```
 10 2000           *=$2000
 20 2000 200000           JSR SETUP
 30 2003 9900D0 CLS       STA $D000,Y
 40 2006 9900D1           STA $D100,Y
 50 2009 9900D2           STA $D200,Y
 60 200C 9900D3           STA $D300,Y
 70 200F 88               DEY
 80 2010 D0F1             BNE CLS
 90 2012 4C0020           JMP $2000
100 2015 A932    SETUP    LDA #$32
110 2017 A000             LDY #$00
120 2019 60               RTS
```

EXAMPLE 1- An "A" Output

Let's say you do not have enough memory to assemble the above code. This requires that the source be broken into two sub-programs that can be assembled one after the other.

Break up the text into two sub-programs between lines 40 and 50. Line 30 has a lable called "CLS". This lable is called in line 80 of the second sub-program (The BNE CLS).

It is necessary to find out what labels are needed for each ub-program(There can be more than 2 sub-programs). This is done by saving the 2nd half of the program (lines 50-120) on tape or disk, and then deleting those lines from the source in memory. Do an A1 on the remaining source. All of the error 18's are the labels you need to define.

When you A1 the first sub-program, you will get an error 18 in line 20. This is because the label "SETUP" is not defined in the symbol table.

Now save the first half. INIZ and load in the saved 2nd sub-program.

The second sub-program will give an error 18 for line 80. The new listing has no "CLS" label in it's symbol table.

If you do an A command to the assembler when the program is still together, it will print out the memory location for all the labels. (Like Example 1.)

"CLS" has a memory location of $2003 & "SETUP" is $2015. It is now necessary to define the labels in the sub-program where they are used.

Now the first half of the program will know where the second half label(s) are, and vice-versa.

To assemble into memory, you must:

(1) Load in the first sub-program
(2) Define the labels from sub-program 2 and renumber.
(3) If you want, save the source (It saves typing)
(4) A3
(5) INIZ
(6) Load in sub-program 2
(7) Define the labels from sub-program 1 and renumber.
(8) If you want, save the source
(9) A3
(10) (sigh)

Here are the listings with the labels defined:

SUB-PROGRAM 1

```
10 2000           *=$2000
20 2000 SETUP=$2015
30 2000 201520           JSR SETUP
40 2003 9900D0 CLS       STA $D000,Y
```

SUB-PROGRAM 2

```
 10 2006           *=$2006
    ^^^^ THIS IS THE START OF THE
    SECOND HALF!!!!
 20 CLS=$2003
 30 2006 9900D1           STA $D100,Y
 40 2009 9900D2           STA $D200,Y
 50 200C 9900D3           STA $D300,Y
 60 200F 88               DEY
 70 2010 D0F1             BNE CLS
 80 2012 4C0020           JMP $2000
 90 2015 A932    SETUP    LDA #32
100 2017 A000             LDY #00
110 2019 60               RTS
```

Note these listings have been renumbered

If you have any questions about this, give me a call or write a letter.

PASSWORD

by Gary Wheeler

When I first received my dual 8" floppies complete with 65D and 65U, I was attracted to 65U because of the ease of programming 65U offers by taking care of all the messy disk management operations. One feature of 65U that particularly caught my eye was the password option that appeared in the CREATE program. Well, needless to say, my programs soon had passwords ranging from AAAA to 1234 to anything else I could dream up. Fortunately, I very soon learned the error of that philosophy and renamed my programs either without a password or with PASS for a password. I fear, however, that some disk users might not be so fortunate. Therefore, I have written this article in the hopes that they may save some software that might otherwise be lost. When one starts to search for a solution to this problem, he is quite quickly led to the disk directory because almost all of the pertinent information about the make-up of a file (including the password) is found here. Each file made has a tag in the directory (DIREC*) which tells its name, password, access rights, disk address, and length. So, if you look at the directory through CHANGE, you will find the file encoded in sixteen bytes in a format similar to this:

CREATE FFF5 06 740000 200000 F0

"CREATE" appears as its ASCII equivalent. FFF5 is the encoded password, in this case "PASS". 06 describes the access rights. Other numbers from 00 to 0B might appear here depending on whether the file is DATA, BASIC, or OTHER and whether its access rights are NONE, READ, WRITE, or R/W. The next three bytes represent in hex the disk address of the file. The next three show the file length, again in hex. The final byte is the header pointer.

The average user needn't worry much about much of this information, as long as he doesn't forget his password. In the event that this should happen, I have written the following program. It makes use of the user programmable disk and access rights of your file. It eliminates the password of your file and leaves it with R/W access rights. I hope this article and program might help some OSI users who might otherwise be lost.

```
10  REM PASSWORD BUSTER PROGRAM  BY GARY
WHEELER
20  TU=PEEK(9832):INPUT"Disk drive";
UN$:DEVUN$
30 POKE8778,192:POKE8779,36:
POKE9432,243:POKE9433,40
40 POKE9435,232:POKE9436,40:
RA=9970:CB=9889:NB=256
50 A=9899:EA=256*(PEEK(A)+256*
(PEEK(A+1)+256*PEEK(A+2)))
60  S=9902:ES=256*(PEEK(S)+256*
(PEEK(S+1)+256*PEEK(S+2)))
70 EN=EA+ES
80  INPUT"Program  name";NA$:
N=LEN(NA$):IFN<1ORN>6THEN80
90 IFN<6THENFORI=0TO6-N:
NA$=NA$+"":NEXT
100 FORI=1TO6:NC(I)=ASC
(MID$(NA$,I,1)):NEXT
110 OF=32
120 DA=EA:GOSUB240
130 RT=RA+OF:IFPEEK(RT)=0THENRUN
140 FORI=0TO5:IFPEEK(RT+I)<>
NC(I+1)THEN210
150 NEXT
160  POKERT+6,0:POKERT+7,0:
PK=INT(PEEK(RT+8)/4)
170 IFPK=0THENPOKERT+8,3
180 IFPK=1THENPOKERT+8,7
190 IFPK=2THENPOKERT+8,11
200 RW=1:GOSUB240:GOTO340
210 OF=OF+16:IFOF<256THEN130
220 EA=EA+256:OF=0:IFEA<ENTHEN120
230 EA=EN:PRINT"File not found"
240  DH=INT(DA/16777216):
RM=DA-DH*16777216
250 DM=INT(RM/65536):    RM=RM-DM*65536
260 DL=INT(RM/256):        RM=RM-DL*256:
DB=RM
270  POKECB+1,DB:POKECB+2,DL:
POKECB+3,DM:POKECB+4,DH
280  POKECB+5,NB-INT(NB/256)*256:
POKECB+6,NB/256
290  POKECB+7,RA-INT(RA/256)*256:
POKECB+8,RA/256
300 ER=USR(RW):IFER=0THENRETURN
310 RW$="READ":IFRW=1THENRW$="WRITE"
320 T=PEEK(9832):IFT>127THENT=T-128+4
330 PRINT:PRINT:PRINT"DEVICE"CHR$(65+T);
RW$"ERROR"ER"AT ADDRESS";DA
340  IFTU>127THENTU=TU-124:
IFTU>63THENTU=TU-58
350 DEV CHR$(65+TU)
360 POKE8778,208:POKE8779,16:END

OK
```

BOB GLICKSMAN, SAN JOSE, CALIFORNIA

I discovered one typographical error in the otherwise excellent DUMB TERMINAL PROGRAM by Daniel Wolf (Aardvark Journal Vol.2, no. 5, page 5, Dec. 1981). The entry at location 1FD0 should be 29 in lieu of 2A. I hope that others among your subscribers will now be spared the intermidable distress that this error caused me.

```
1 REM*DUMB TERMINAL PROGRAM
2 REM*SET MEMORY < 7000
3 REM*OR THIS PROGRAM WON'T WORK!!!!
10  FORA=0TO192:B=PEEK(64768+A):
POKE7936+A,B:NEXT
20 REM PATCHES:AD=8191
60 READC
65 IFC=10000THEN100
70 IFC<0THENAD=-C:GOTO60
80 POKEAD,C
85 AD=AD+1
90 GOTO60
100 POKE11,0:POKE12,31
105 FORI=1TO30:PRINT:NEXT:  PRINT"DUMB
TERMINAL EMULATION"
110 X=USR(X)
1000 DATA-8052 , 79
1010 DATA-8129 , 76 , 216 , 31
1020 DATA-8132 , 173 , 0 , 240
1030 DATA-8135 , 74
1040 DATA-8136 , 144
1050 DATA-8137 , 11
1060 DATA-8138 , 173 , 1 , 240
1070 DATA-8141 , 234 , 234 , 234
1080 DATA-8144 , 41 , 127
1090 DATA-8146 , 32 , 45 , 191
1100 DATA-8149 , 76 , 4 , 31
1110 DATA-8152 , 32 , 177 , 252
1120 DATA-8155 , 169 , 0
1130 DATA-8157 , 234 , 234 , 234
1140 DATA-8160 , 76 , 4 , 31
1150 DATA10000
OK
```

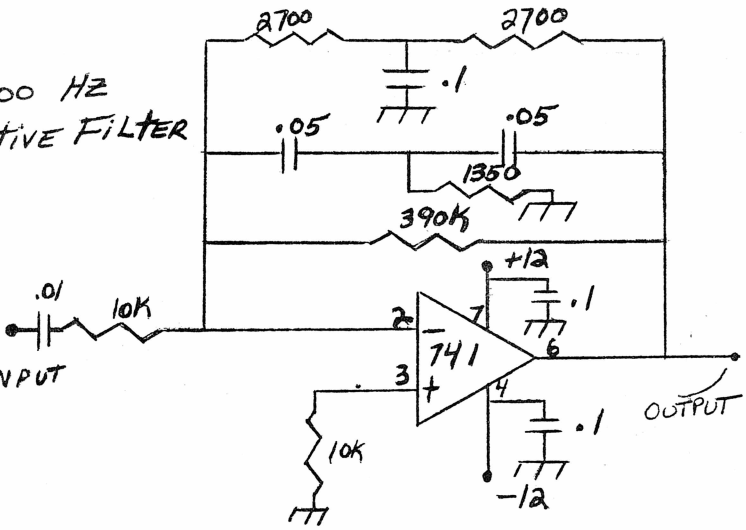
PAT McCANN, BATTLE CREEK, MICHIGAN

This program was written for use by a blind person and as such replaces normal output. Anything that would normally be printed on the ourput scrolling line will be hear as Morse Code. I could not find any CW reprsentations for the following: space, !, #, %, found for these characters there is space set aside in the tables for them. The program is in BASIC andpokes a machine language program into the top of 8K. It takes about 256 bytes. If the user has more or less memory it can be relocated. I have included an assembler listing and output to make this easy.

The computer can run all normal functions except "SAVE" and allow time for the CW code. Because of this to use "SAVE" or "LOAD" one must do a warm start, to get back the normal output routine. To reenter CW output POKE 538,0 and POKE 539,31.

The CW is output through the tape port using the 6850. There are two ways to use this signal. One is to listen to the tape signal going to the recorder after running it through a filter to get rid of the 2400 cycle tone. This can be a very simple circuit such as this:

1200 Hz Active Filter



The other ways are to key and audio ocillator using one of the following outputs: RS232 or TTL if implemented or to bring out the signal from U-18 pin 10 or 12 depending on what polarity is needed to key oscillator.

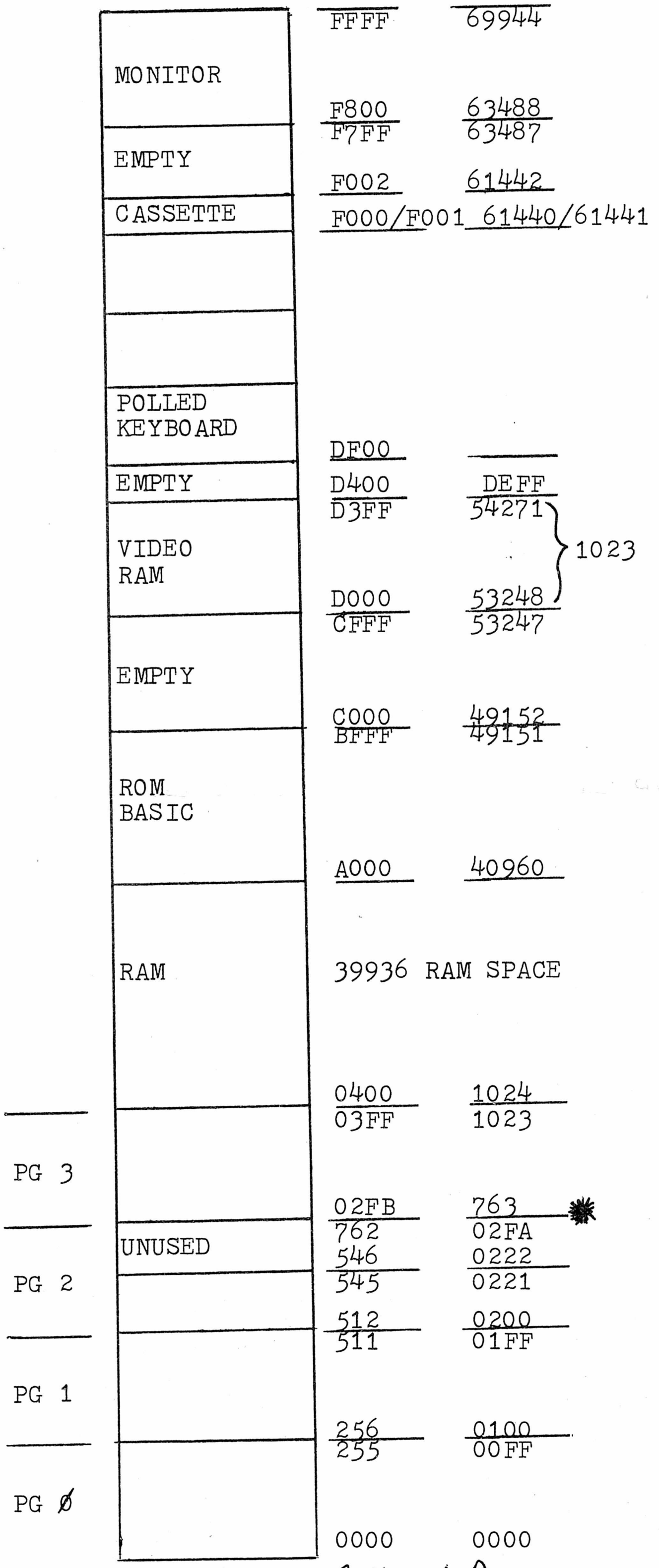
```
10 REM BY PAT MCCAN
12 BATTLE CREEK MI
50 REM THIS SUB POKES  CW  OUTPUT INTO
MACHINE LANGUAGE
51 REM FOR  VECTOR PUT 0 INTO 538 & 31
INTO 539
52 FOR IN=7936TO8191
54 READ OP :POKE IN,OP
56 NEXT IN
60 DATA8,72,133,247,138,72,152
70 DATA72,165,247,32,105,255,166
80 DATA247,224,32,176,3,76,76
90 DATA31,224,91,144,3,76,76
100 DATA31,188,148,31,240,81,189
110 DATA89,31,133,243,165,243,16
120 DATA24,162,3,169,255,141,0
130 DATA240,32,86,31,202,208,245
140 DATA169,0,141,0,240,32,86
150 DATA31,76,71,31,162,1,208
160 DATA230,6,243,136,208,219,32
170 DATA106,31,104,168,104,170,104
180 DATA40,96,134,245,162,32,169
190 DATA127,133,241,198,241,165,241
200 DATA208,250,202,208,243,166,245
210 DATA96,162,4,32,86,31,202
220 DATA208,250,96,32,106,31,76
230 DATA76,31,0,0,72,0,0
240 DATA0,0,120,180,180,152,100
250 DATA204,132,04,144,248,120,56
260 DATA24,8,0,128,192,224,240
270 DATA224,168,76,136,216,48,0
280 DATA64,128,160,128,0,32,192
290 DATA0,0,112,160,64,192,128
300 DATA224,96,208,64,0,128,32
310 DATA16,96,144,176,192,0,0
320 DATA6,0,0,0,0,6,6
330 DATA6,6,8,6,6,6,5
340 DATA5,5,5,5,5,5,5
350 DATA5,5,5,6,6,7,5
360 DATA6,6,0,2,4,4,3
370 DATA1,4,3,4,2,4,3
380 DATA4,2,2,3,4,4,3
390 DATA3,1,3,4,3,4,4
400 DATA4,58,136,44,244,0,42
410 DATA146,0,25,243,0,106,102
420 DATA0,125,1,240
450 POKE538,0:POKE539,31
455 POKE134,31
460 PRINT" CW OUTPUT READY"
470 PRINT"TO CHANGE SPEED POKE NEW SPEED
INTO 8025"
480 PRINT"TO USE TAPE HIT BREAK AND DO A
WARM START"
490 PRINT"TO REENTER CW  OUTPUT POKE
538,0 AND 539,31"
500  PRINT"TYPE  NEW  TO  ERASE BASIC
PROGRAM LEAVING MACHINE CODE"
OK
```

<u>CORRECTION by Charles Schall</u> from previous article by him in December 1981 Vol. 5, #5, page 8 of Aardvark journal on Extended Monitor. Line 65 should read as follows:

```
65 IFA=185THENGOSUB100
```

MEMORY MAP

by Bobby Akins



NOTE: SWAP of HEX & DEC ✱ CODES FOR EASE of HANDLING.

WE JUST SENT $475 IN GIFT CERITIFICATES TO THE DECEMBER AUTHORS/E2 AND $600 IN CERTIFICATES TO THE FEBRUARY AUTHORS. GET YOUR SHARE BY SENDING AN ARTICLE, IDEA OR PARAGRAPH TO THE JOURNAL

RODGER OLSEN

FOR SALE: OSI C4P SERIES II, 8K cassette, 16 colors, sound, in original box w/many games, OSI primer. Aardvark journal and other info. $550.00. Call Don (313) 886-7303

DAVID BROYHILL, RINGGOLD, GEORGIA

BAUD RATE SELECTOR

A 74LS151 was used to give a BCD selection of baud rate from 9600 to 300. Three switches coule be used however to have software control two switches andthe bRTS line were used to select the baud rate. The input frequencies can be selected to give the desired software selection - POKE61440,85 for a high and 61440,25 for a low. POKE61440,25 was used because a 0 disabled the load in my C1P.

If you have added the switch to double your baud rate, then you will have a greater selection. The 74LS151 was added to the proto area of the board and the only cut necessary is at pin 2 on U55. C4 was cut and connected to Pin 5 on the 74LS151. C0,C1,C2,C3,C4, and C5 were jumpered to the inputs of the new chip and other connections were made as shown. Pick the baud rate you desire to switch between so they will be selected by the bRTS on Pin 11.

PRINTER SELECTOR

To give complete software selection of the printer a new chip - 74LS02 was added to the proto board and a 74LS04 was added in the U68 location. The U68 calls for a 75LS07, but I had the 04 so it was used instead. The line between U62 pin 8 and J3 terminal 10 was cut and the nor was added in the line. One gate is controlled by RTS not and the other by the printer handshake through Q2 and U62. If POKE61440,85 is given, RTS not goes low and the printer handshake has control of the CTSNOT line. When 25 is poked into location 61440 then RTSNOT goes high and CTSNOT will be forced low regardless of the printer handshake. The only other cut necessary is on pin 24 of U14. Cut the ground jumped and add a jumper to CTS not

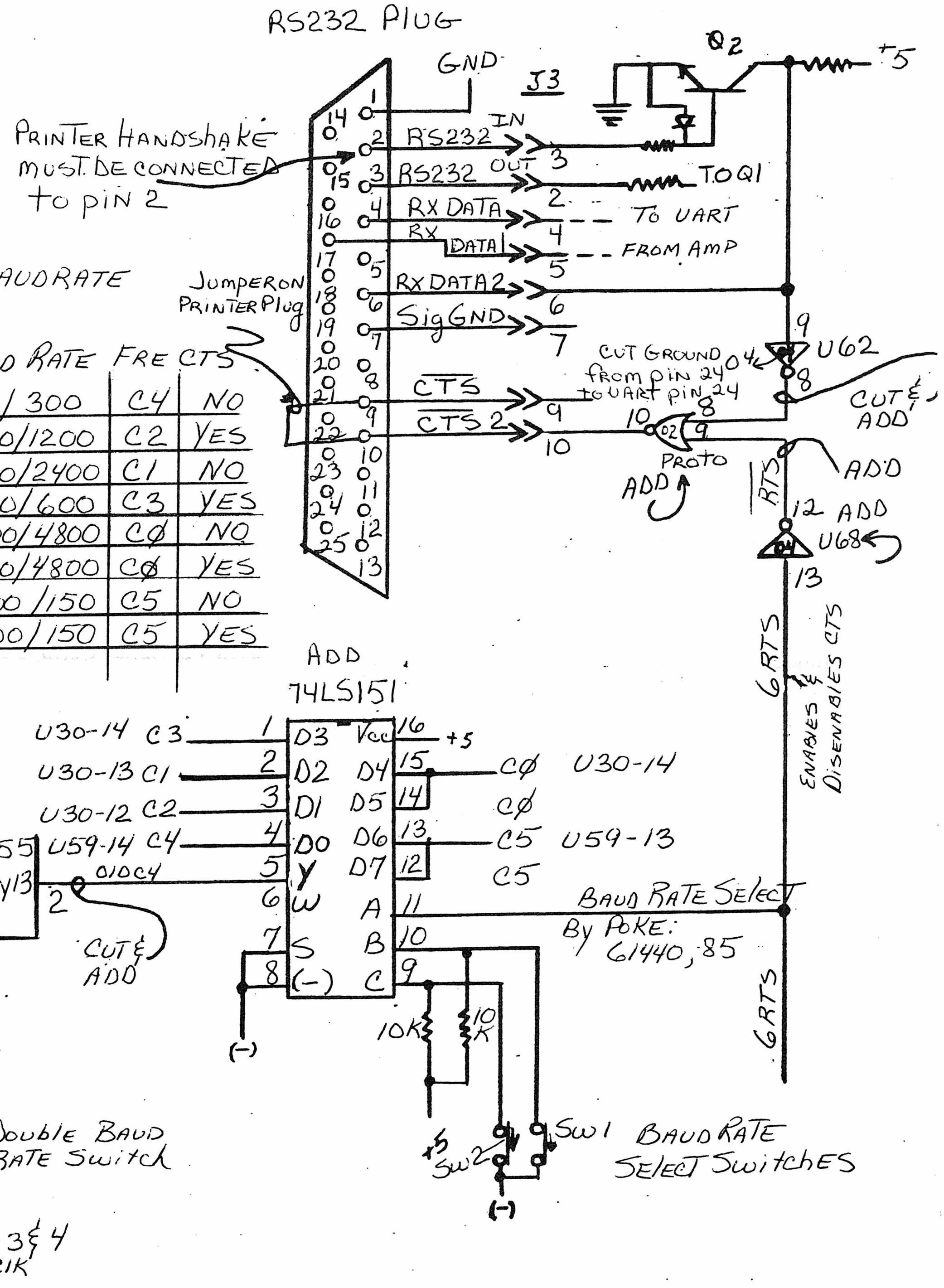
X=85 For H
X=25 For L

RX DATA LINES
ADDED FOR FUTURE USE

Poke 61440,X
SW1
SW2

BAUDRATE

A	B	C	BAUD RATE	FRE	CTS
L	L	L	600/300	C4	NO
H	L	L	2400/1200	C2	YES
L	H	L	4800/2400	C1	NO
H	H	L	1200/600	C3	YES
L	L	H	9600/4800	C0	NO
H	L	H	9600/4800	C0	YES
L	H	H	300/150	C5	NO
H	H	H	300/150	C5	YES



YACHT RACE

```
 10 REM JACK VAUGHN
 19 REM LINES 20-28 SET UP MC SCREEN CLE
AR (X=USR(X))
 20 FORX=576TO598:READY:POKEX,Y:NEXT
 22 DATA169,208,160,0,133,255,132,254
 24 DATA162,4,169,32,145,254,200,208,251
 26 DATA230,255,202,208,246,96
 28 POKE11,64:POKE12,2
 30 X=USR(X)
 35 VIDEO=600:IFPEEK(57088)<129THENVIDEO
=540
 36 L=32:CO=53446:REM C1 CORNER AND LINE
LENGTH
 38 IFVI=600GOTO49
 40 L=64:CO=53636:REM C2 VALUES
 49 JA=226:JB=219:JC=188:JD=45:REM SCREE
N SYMBOLS
 50 REM COMMITTEE BOAT SYMBOLS & LOC.
 51 B=193:H=161:CL=CO+L*11+17
 53 REM DISQUALIFIED LOCATIONS
 55 AQ=CO+L*23+10:BQ=CO+L*23+14:CQ=CO+L*
23+18
 62 REM MARK SYMBOLS AND LOCATIONS
 65 M=111:MA=CO+L*11+12:MB=CO+L*3+14:MC=
CO+L*11+3:MD=CO+L*19+14
 68 REM KEEP TRACK OF TIME       ACROSS S
TART/FINISH LINE
 70 WA=0:WB=0:WC=0
 74 REM INITIAL LOCATIONS
 75 SA=CO+L*17+10:SB=CO+L*17+14:SC=CO+L*
17+18
 76 REM STARTING LINE & TURN #
 77 SL=MA+1:TU=0
 78 REM MN IS MOVE AMOUNT.       PN IS PE
NALTY AMOUNT
 80 MN(1)=L+1:MN(2)=L-1:MN(3)=L:MN(4)=2:
IFVI=600THENMN(4)=1
 84 PN(1)=L*3:PN(2)=3*(L-6):PN(3)=3*(L-3
):PN(4)=INT(L/10):GOTO300
 100 X=USR(X):POKEMA,M:POKEMB,M:POKEMC,M
 101 POKEMA,1,65:POKEMB+32,66:POKEMC+1,6
7:POKEMD-32,68
 102 POKEMD,M:POKECL,H:POKECL-32,B
 110 POKESL,JD:POKESL+1,JD:POKESL+2,JD:P
OKESL+3,JD:CH=CO
 115 POKECH,78:POKECH+1,87:POKECH+3,78:P
OKECH+5,78:POKECH+6,69
 120 POKECH+L,87:POKECH+L+3,48:POKECH+L+
6,69:POKECH+L+L,83
 125 POKECH+L+L+1,87:POKECH+L+L+3,83:POK
ECH+L+L+5,83:POKECH+L+L+6,69
 148 PA=PEEK(SA):PB=PEEK(SB):PC=PEEK(SC)
:POKESA-AM,32:POKESB-BM,32
 150 POKESC-CM,32:POKESC,JC:POKESB,JB:PO
KESA,JA
 190 RETURN
 300 X=USR(X)
 305 PRINT"      YACHT RACE ?"
 306 FORX=1TO9:PRINT:NEXT
 400 FORX=1TO2000:NEXT:INPUT" INSTRUCTIO
NS ";Y$
 402 IFLEFT$(Y$,1)="Y"THENGOTO3000
 403 FORX=1TO8:PRINT:NEXT
 405 INPUT" 2 OR 3 PLAYERS";NP
 411 FORX=1TO5:PRINT:NEXT
 415 X=USR(X)
 455 CP$="  COLLISION PENALTY"
 460 INPUT" SKIPPER A'S NAME";SA$
 465 PRINT:PRINT
 470 PRINT" CAPTAIN "SA$"'S":PRINT" SYMB
OL IS "CHR$(JA)
 475 FORX=1TO5:PRINT:NEXT
 480 INPUT" SKIPPER B'S NAME";SB$
 485 PRINT:PRINT
 490 PRINT" CAPTAIN "SB$"'S":PRINT" SYMB
OL IS "CHR$(JB)
 492 IFNP=2GOTO515
 495 FORX=1TO5:PRINT:NEXT
 500 INPUT" SKIPPER C'S NAME";SC$
 505 PRINT:PRINT
 510 PRINT" CAPTAIN "SC$"'S":PRINT" SYMB
OL IS "CHR$(JC)
```

```
 515 PRINT:PRINT:PRINT
 520 INPUT" READY";A$
 525 GOSUB100
 530 IFSA=AQTHENGOSUB2300
 535 PRINT" CAPTAIN":PRINT" "SA$"'S":INP
UT" MOVE ";AM$
 536 AM=44:IFAM$="NW"THENAM=-MN(1)
 537 IFAM$="N"THENAM=0
 538 IFAM$="NE"THENAM=-MN(2)
 539 IFAM$="W"THENAM=-MN(4)
 540 IFAM$="O"THENAM=0
 541 IFAM$="E"THENAM=MN(4)
 542 IFAM$="SW"THENAM=MN(2)
 543 IFAM$="S"THENAM=MN(3)
 544 IFAM$="SE"THENAM=MN(1)
 547 SA=SA+AM:IFAM=44GOTO594
 549 RN=INT(RND(8)*100):TU=TU+1:IFTU=RNT
HENTU$=SA$:GOSUB6000
 550 IFTU=RN+2THENTU$=SA$:GOSUB7000
 551 GOSUB1200:GOSUB100
 552 IFPA=JDTHENIFWA=1THENW$=SA$:GOTO240
0
 553 IFPA=JDTHENWA=1
 554 GOTO600
 594 FORX=1TO10:PRINT:NEXT
 595 SA=SA-AM:PRINT" ILLEGAL MOVE "AM$:A
M=0:PRINT:GOTO530
 600 IFSB=BQTHENGOSUB2300
 605 PRINT" CAPTAIN":PRINT" "SB$"'S":INP
UT" MOVE ";BM$
 606 BM=44:IFBM$="NW"THENBM=-MN(1)
 607 IFBM$="N"THENBM=0
 608 IFBM$="NE"THENBM=-MN(2)
 609 IFBM$="W"THENBM=-MN(4)
 610 IFBM$="O"THENBM=0
 611 IFBM$="E"THENBM=MN(4)
 612 IFBM$="SW"THENBM=MN(2)
 613 IFBM$="S"THENBM=MN(3)
 614 IFBM$="SE"THENBM=MN(1)
 615 SB=SB+BM:IFBM=44GOTO659
 617 IFTU=RN+2THENTU$=SB$:GOSUB7000
 618 RN=INT(RND(8)*100):TU=TU+1:IFTU=RNT
HENTU$=SB$:GOSUB6000
 619 GOSUB1300:GOSUB100
 620 IFPB=JDTHENIFWB=1THENW$=SB$:GOTO240
0
 625 IFPB=JDTHENWB=1
 655 GOTO698
 659 FORX=1TO10:PRINT:NEXT
 660 SB=SB-BM:BM=0:PRINT"  ILLEGAL MOVE
"BM$:PRINT:GOTO600
 698 IFNP=2THENSC=CQ:GOTO530
 700 CM=44:IFSC=CQTHENGOSUB2300
 702 PRINT" CAPTAIN":PRINT" "SC$"'S":INP
UT" MOVE ";CM$
 703 IFCM$="NW"THENCM=-MN(1)
 704 IFCM$="N"THENCM=0
 705 IFCM$="NE"THENCM=-MN(2)
 706 IFCM$="W"THENCM=-MN(4)
 707 IFCM$="O"THENCM=0
 708 IFCM$="E"THENCM=MN(4)
 709 IFCM$="SW"THENCM=MN(2)
 710 IFCM$="S"THENCM=MN(3)
 711 IFCM$="SE"THENCM=MN(1)
 714 SC=SC+CM:IFCM=44GOTO755
 716 RN=INT(RND(8)*100):TU=TU+1:IFTU=RNT
HENTU$=SC$:GOSUB6000
 717 IFTU=RN+2THENTU$=SC$:GOSUB7000
 718 GOSUB1400:GOSUB100
 720 IFPC=JDTHENIFWC=1THENW$=SC$:GOTO240
0
 725 IFPC=JDTHENWC=1
 752 GOTO530
 755 FORX=1TO10:PRINT:NEXT
 760 SC=SC-CM:CM=0:PRINT"  ILLEGAL MOVE
"CM$:PRINT:GOTO700
 790 GOTO530
 1200 IFSA=MDTHENSA=SA-4:GOTO1220
 1202 IFSA=MCTHENSA=SA+4:GOTO1220
 1204 IFSA=MBTHENSA=SA+PN(1):GOTO1220
 1206 IFSA=MATHENSA=SA+PN(1):GOTO1220
 1208 IFSA=CLTHENSA=SA+PN(1):GOTO1220
 1209 IFSA=CL-32THENSA=SA+PN(1):WA=0:GOT
01220
 1210 IFSA=SBTHENFORX=1TO32:PRINTCP$:NEX
T
 1211 IFSA=SCTHENFORX=1TO32:PRINTCP$:NEX
T
 1213 IFSA<>SCTHENIFSA<>SBTHENRETURN
 1214 IFSA=SBTHENFORX=1TO6:PRINT"  CAP."
SA$:PRINT"  DISQUALIFIED":SA=AQ
 1216 IFSA=SCTHENFORX=1TO6:PRINT" CAP."S
A$:PRINT"  DISQUALIFIED":SA=AQ
 1218 FORX=1TO2000:NEXT:RETURN
 1220 FORX=1TO5:PRINTCP$:NEXT:FORX=1TO20
00:NEXT:RETURN
 1300 IFSB=MDTHENSB=SB-4:GOTO1320
 1302 IFSB=MCTHENSB=SB+4:GOTO1320
 1304 IFSB=MBTHENSB=SB+PN(1):GOTO1320
 1306 IFSB=MATHENSB=SB+PN(1):GOTO1320
 1308 IFSB=CLTHENSB=SB+PN(1):GOTO1320
 1309 IFSB=CL-32THENSB=SB+PN(1):WB=0:GOT
01320
 1310 IFSB=SATHENFORX=1TO32:PRINTCP$:NEX
T
 1311 IFSB=SCTHENFORX=1TO32:PRINTCP$:NEX
T
 1312 IFSB<>SATHENIFSB<>SCTHENRETURN
 1313 IFSB=SCTHENFORX=1TO6:PRINT" CAP."S
B$" DISQUALIFIED":NEXT:SB=BQ
 1314 IFSB=SATHENFORX=1TO6:PRINT" CAP."S
B$" DISQUALIFIED":NEXT:SB=BQ
 1316 IFSB=SCTHENFORX=1TO6:PRINT" CAP."S
B$" DISQUALIFIED":NEXT:SB=BQ
 1318 FORX=1TO1500:NEXT:RETURN
 1320 FORX=1TO5:PRINTCP$:NEXT:FORX=1TO20
00:NEXT:RETURN
 1400 IFSC=MDTHENSC=SC-4:GOTO1420
 1402 IFSC=MCTHENSC=SC+4:GOTO1420
 1404 IFSC=MBTHENSC=SC+PN(1):GOTO1420
 1406 IFSC=MATHENSC=SC+PN(1):GOTO1420
 1408 IFSC=CLTHENSC=SC+PN(1):GOTO1420
 1409 IFSC=CL-32THENSC=SC+PN(1):WC=0:GOT
01420
 1410 IFSC=SATHENFORX=1TO32:PRINTCP$:NEX
T
 1411 IFSC=SBTHENFORX=1TO32:PRINTCP$:NEX
T
 1413 IFSC<>SATHENIFSC<>SBTHENRETURN
 1414 IFSC=SATHENFORX=1TO6:PRINT"  "SC$"
 DISQUALIFIED":NEXT:SC=CQ
 1416 IFSC=SBTHENFORX=1TO6:PRINT" "SC$"
DISQUALIFIED":NEXT:SC=CQ
 1418 FORX=1TO1600:NEXT:RETURN
 1420 FORX=1TO5:PRINTCP$:NEXT:FORX=1TO20
00:NEXT:RETURN
 2300 IFSA=AQTHENIFSB<>BQTHENIFSC<>CQGOT
O600
 2302 IFSA=AQTHENIFSB<>BQTHENIFSC=CQTHEN
W$=SB$:GOTO2400
 2304 IFSA<>AQTHENIFSB=BQTHENIFSC=CQTHEN
W$=SA$:GOTO2400
 2306 IFSA<>AQTHENIFSB=BQTHENIFSC<>CQGOT
O700
 2308 IFSA=AQTHENIFSB=BQTHENIFSC<>CQTHEN
W$=SC$:GOTO2400
 2310 GOTO530
 2400 PRINT"************************
 2402 PRINT
 2404 PRINT"*** THE WINNER IS - ***"
 2405 PRINT:PRINT
 2406 FORX=1TO3:PRINT"******************
******":NEXT
 2407 FORX=1TO1800:NEXT
 2408 FORX=1TO30:PRINT"*****************
*******":NEXT
```

```
 2410 PRINT
 2412 PRINT"  THREE CHEERS FOR
 2414 PRINT
 2416 PRINT"   CAPTAIN "W$
 2417 PRINT: FORX=1TO6:PRINT"***********
*************
 2418 NEXT:PRINT
 2420 INPUT" PLAY AGAIN ";A$
 2422 IFLEFT$(A$,1)="Y"GOTO30
 2425 PRINT:PRINT"  THANK YOU":PRINT:PRI
NT"  END":END
 3000 PRINT:PRINT" YOU CAN MOVE ANY":PRI
NT:PRINT" DIRECTION ON CHART"
 3002 PRINT:PRINT" IN UPPER LEFT COR-":P
RINT:PRINT" NER.ENTER MOVE AND"
 3003 PRINT:PRINT" PRESS RETURN. . . .":
PRINT
 3004 PRINT" PENALTIES ARE GIVEN":PRINT:
PRINT" FOR  HITTING A MARK
 3006 PRINT:PRINT" OR  FOR HITTING THE":
PRINT:PRINT" STARTING BOAT.
 3008 PRINT:INPUT" MORE";A$
 3010 PRINT:PRINT" YOU WILL BE OUT OF":P
RINT:PRINT" THE GAME IF YOU HIT
 3012 PRINT:PRINT" ANOTHER RACING YACHT.
":PRINT
 3014 PRINT" THE OBJECT IS TO":PRINT:PRI
NT" CROSS STARTING LINE TO
 3016 PRINT:PRINT" RIGHT OF MARK 'A',":P
RINT:PRINT" THEN CONTINUE TO
 3018 PRINT:PRINT" ROUND MARKS 'B','C',"
:PRINT:PRINT" AND 'D'. YOU MUST
 3019 PRINT
 3020 PRINT" CROSS THE FINISH":PRINT:PRI
NT" LINE FIRST TO WIN.
 3029 PRINT:PRINT:PRINT
 3030 INPUT" GO ON ";A$
 3032 PRINT
 3035 PRINT" IF YOU GET OUT OF THE
 3037 PRINT:PRINT" PROGRAM, TYPE IN -
 3040 PRINT:PRINT" GO TO 535 (FIRST)
 3042 PRINT:PRINT" OR- GO TO 600 FOR
 3045 PRINT:PRINT" THE SECOND PLAYER
 3046 PRINT:PRINT
 3047 INPUT" READY TO START";A$
 3048 PRINT:PRINT
 3050 GOTO405
 6000 PRINT"   HIGH WINDS!!":PRINT:PRINT
"   CAPTAIN "TU$:PRINT
 6002 PRINT"   HAS BROACHED AND HAS":PRI
NT:PRINT"   BEEN DELAYED
 6004 IFTU$=SA$THENSA=SA+L*3
 6006 IFTU$=SB$THENSB=SB+L*3
 6008 IFTU$=SC$THENSC=SC+L*3
 6009 FORX=1TO1800:NEXT
 6010 PRINT:RETURN
 7000 XZ=0
 7001 X=USR(X):FORI=1TO7:PRINT:NEXT:PRIN
T"    MAN OVERBOARD"
 7002 FORI=1TO250:NEXT:GOSUB100:XZ=XZ+1:
FORI=1TO125:NEXT
 7003 FORI=1TO7:PRINT:NEXT:IFXZ=7GOTO700
5
 7004 GOTO7001
 7005 X=USR(X):PRINT" CAPTAIN "TU$:PRINT
:PRINT" HAS MAN OVERBOARD"
 7006 FORI=1TO1000:NEXT
 7008 IFTU$=SA$THENSA=SA+L*3
 7010 IFTU$=SB$THENSB=SB+L*3
 7011 IFTU$=SC$THENSC=SC+L*3
 7015 FORX=1TO1500:NEXT:RETURN
```

JOURNAL SUBSCRIPTION FORM

NAME ______________________________

ADDRESS ______________________________

CITY ______________________________

STATE ______________________ ZIP CODE ______________________

( ) CHECK ( ) VISA CARD # ______________________

( ) MONEY ORDER ( ) MASTER CHARGE EXP. DATE ______________

☐ RENEWAL (VOL.3)
APR. 82 - FEB. 83
$ 14.00

☐ CAN'T KEEP IT TO MYSELF - SEND MY FRIEND A SUBSCRIPTION

☐ NEW-CHECK APPROPRIATE BOX

☐ VOL. #1 - I want all the back issues. APR. 80 - FEB. 81 - $9.00

☐ VOL. #2 - I want all the back issues. APR. 81 - FEB. 82 - $9.00

☐ I LIKE TO READ - SEND BOTH VOL. #1 and VOL. # 2. $18.00

NOTE ↓

OVERSEAS SUBSCRIPTIONS FOR VOL. #1 & VOL. # 2 $14.00
OVERSEAS SUBSCRIPTIONS FOR VOL. #3 $19.00

AARDVARK
2352 SOUTH COMMERCE
WALLED LAKE, MI 48088

BULK RATE
U. S. POSTAGE
PAID
WALLED LK., MI.
Permit No. 5

J. COOK
402 N. SHORE DR.
BELLINGHAM
WA, 98226